An inexact dual logarithmic barrier method for solving sparse semidefinite programs

Noname manuscript No.
(will be inserted by the editor)

An inexact dual logarithmic barrier method for solving sparse semidefinite programs

Stefania Bellavia · Jacek Gondzio · Margherita Porcelli

Received: date / Accepted: date

Abstract A dual logarithmic barrier method for solving large, sparse semidefinite programs is proposed in this paper. The method avoids any explicit use of the primal variable X and therefore is well-suited to problems with a sparse dual matrix S. It relies on inexact Newton steps in dual space which are computed by the conjugate gradient method applied to the Schur complement of the reduced KKT system.

The method may take advantage of low-rank representations of matrices A_i to perform implicit matrix-vector products with the Schur complement matrix and to compute only specific parts of this matrix. This allows the construction of the partial Cholesky factorization of the Schur complement matrix which serves as a good preconditioner for it and permits the method to be run in a matrix-free scheme.

Convergence properties of the method are studied and a polynomial complexity result is extended to the case when *inexact* Newton steps are employed.

The work of the first and the third author was supported by *Gruppo Nazionale per il Calcolo Scientifico* (GNCS-INdAM) of Italy. The work of the second author was supported by EPSRC Research Grant EP/N019652/1.

S. Bellavia
Dipartimento di Ingegneria Industriale, Università degli Studi di Firenze, viale Morgagni 40, 50134 Firenze, Italia
E-mail: stefania.bellavia@unifi.it

J. Gondzio
School of Mathematics, The University of Edinburgh, James Clerk Maxwell Building, The Kings Buildings, Peter Guthrie Tait Road, Edinburgh, EH9 3FD, UK
and NASK Research Institute, Kolska 12, 01-045, Warsaw, Poland
E-mail: j.gondzio@ed.ac.uk

M. Porcelli
Dipartimento di Ingegneria Industriale, Università degli Studi di Firenze, viale Morgagni 40, 50134 Firenze, Italia
E-mail: margherita.porcelli@unifi.it

A Matlab-based implementation is developed and preliminary computational results of applying the method to maximum cut and matrix completion problems are reported.

Keywords Semidefinite programming · Dual logarithmic barrier method · Inexact Newton method · Preconditioning

Mathematics Subject Classification (2000) 90C22, 90C51, 65F10, 65F50

1 Introduction

Let $S\mathbb{R}^{n\times n}$ denote the set of real symmetric matrices of order n and let $U \bullet V$ denote the inner product between two matrices, defined by $trace(U^TV)$. Consider the standard semidefinite programming (SDP) problem in its primal form

$$\begin{array}{ll} \min & C \bullet X \\ \text{s.t.} & X \succeq 0 \\ & A_i \bullet X = b_i \quad i = 1, \ldots, m, \end{array} \tag{1}$$

where $A_i, C \in S\mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^m$ are given and $X \in S\mathbb{R}^{n\times n}$ is unknown and assume that matrices $A_i, i = 1, 2, \ldots, m$ are linearly independent, that is $\sum_{i=1}^m A_i d_i = 0$ implies $d_i = 0$, $i = 1, \ldots, m$. The dual form of the SDP problem associated with (1) is:

$$\begin{array}{ll} \max & b^T y \\ \text{s.t.} & \sum_{i=1}^m y_i A_i + S = C \\ & S \succeq 0, \end{array} \tag{2}$$

where $y \in \mathbb{R}^m$ and $S \in S\mathbb{R}^{n\times n}$.

In this paper we are concerned with the solution of problems where the dual variable S is very sparse. Such situations arise when matrices A_i, $i = 1, \ldots, m$ and C share the sparsity patterns [25], and are common in relaxations of optimization problems such as, e.g. maximum cut and matrix completion problems [3,5].

Semidefinite programming is a well established area of convex optimization [8,21,26]. Over the last two decades many powerful techniques have been developed for the solution of SDP problems. Although the majority of developments in this area relied on interior point methods, there have been also successful attempts to employ different techniques such as a specialized variant of bundle method [14], augmented Lagrangian approach [27] or modified barrier method [15].

Interior point methods for SDP have an advantage: they have provable low worst-case iteration complexity [8,21]. On the other hand, the solution of real-life SDPs still remains a computational challenge because the linear systems involved in interior point methods for SDP have dimensions $n^2 + m$ or m for augmented system or Schur complement, respectively. Such systems

may be prohibitive for any larger values of n and m. Most of standard IPM implementations work with the $m \times m$ Schur complement linear system. For larger values of m building, storing and inverting this matrix is still a major challenge. There have been of course several attempts to overcome these difficulties. They usually rely on an application of Krylov subspace methods for solving the linear equations resulting from the reduced KKT systems [22,23].

The challenge originates from the complexity of the reduced KKT systems which are large, involve products of matrices and often produce dense matrices of very large dimension. In the large-scale setting, direct methods of linear algebra are not an option. Iterative methods have to be employed. They are efficient in the early stage of Interior Point procedures, but they struggle in the late stage due to ill conditioning of matrices involved [23].

In quest for a perfect interior point method for SDP one has to compromise between several conflicting objectives. An ideal algorithm would:

- share the best known worst-case iteration complexity,
- have low memory requirements (avoid storing dense matrices of size n or m),
- efficiently compute the Newton direction.

The method presented in this paper is an attempt to satisfy these objectives at least for a wide class of SDP problems which enjoy the property of having sparse dual matrix S.

We propose a dual logarithmic barrier method which maintains only the dual solution of the problem (y, S) and avoids any operations which could involve the primal matrix X (which is likely to be dense). Benson, Ye and Zhang [4] have analysed the dual potential reduction Interior Point method and have demonstrated certain advantages resulting from the ability to avoid using explicit primal matrix X. In a later technical report Choi and Ye [6] mentioned a possibility of employing an iterative linear algebra approach in the context of algorithm [4]. Without providing convergence analysis, authors observed that in this latter approach the algorithm terminates at a primal-dual sub-optimal solution depending on the accuracy imposed on the iterative linear solver. Moreover, they proposed to use a simple diagonal preconditioner for the linear system which naturally had only a very limited ability to improve the spectral properties of the system. An alternative approach has been introduced in [23] in the context of primal-dual method. In this approach in the late steps of the method a decomposition of the Schur complement is performed giving rise to a projected Schur complement to which the iterative method is applied.

The algorithm we propose here makes steps in inexact Newton directions which are computed by an approximate solution of the reduced KKT systems. The system is reduced to the Schur complement form and solved with the preconditioned conjugate gradient method. The Schur complement does not have to be constructed or stored because the CG algorithm needs only to perform matrix-vector multiplications with it and these operations can be executed as a sequence of simple matrix-vector products which involve only very sparse matrices. The procedure is particularly attractive when matrices A_i

are low rank. Krylov-subspace methods are known to benefit from clustering of the spectrum of linear system. Unfortunately, there is no chance for this to happen in the case of systems arising from interior point methods. To improve the spectral properties of the linear system we employ a partial Cholesky preconditioner [1,12].

Much of the effort in the analysis of the proposed approach has gone into designing implementable conditions for the acceptable inexactness in the Newton direction and choosing an appropriate preconditioned iterative method which can meet such conditions. The used preconditioner is compatible with the matrix-free regime of the whole method and still delivers the necessary improvement of the spectral properties of the linear system. We are not aware of any method prior to that one which would meet such conditions, except for the strategies proposed in [22,23] that might provide an inspiration to create a viable alternative.

We also design a short-step variant of the method and show that it enjoys the $\mathcal{O}(\sqrt{n}\ln\frac{n}{\varepsilon})$ worst-case iteration complexity.

The paper is organised as follows. After a brief summary of notation used in SDP, in Section 2 a framework of the dual barrier algorithm is presented. Next, in Section 3 an inexact variant of the method is introduced and some basic facts concerning the proximity of the dual iterates to the central path are discussed. Several technical results needed to establish the convergence of the Inexact Newton method are presented in Section 4. Then a complete analysis of the short-step inexact dual logarithmic barrier method is delivered in Section 5. The methods proposed in this paper have been implemented. The computation of inexact Newton directions employs the preconditioned conjugate gradient algorithm to find an approximate solution of the Schur complement form of the reduced KKT systems. The computation of a preconditioner and the matrix-free implementation of the method are discussed in Section 6. The preliminary computational results obtained with the methods applied to solve the SDP relaxations of maximum cut and matrix completion problems are presented in Section 7 and finally the conclusions are given in Section 8.

Notation. The norm of the matrix associated with the inner product between two matrices $U \bullet V = trace(U^T V)$ is the Frobenius norm, written $\|U\|_F := (U \bullet U)^{1/2}$, while $\|\cdot\|_2$ denotes the operator norm of a matrix. Norms on vectors will always be Euclidean.

Let $\mathcal{A}$ be the linear operator $\mathcal{A} : S\mathbb{R}^{n\times n} \to \mathbb{R}^m$ defined by

$$\mathcal{A}(X) = (A_i \bullet X)_{i=1}^m \in \mathbb{R}^m,$$

then its transposition $\mathcal{A}^T$ is a mapping from $\mathbb{R}^m$ to $S\mathbb{R}^{n\times n}$ given by

$$\mathcal{A}^T v = \sum_{i=1}^m v_i A_i.$$

Moreover, let A^T denote the matrix representation of $\mathcal{A}^T$ with respect to the standard bases of $\mathbb{R}^n$, that is

$$A^T := [vec(A_1), vec(A_2), \ldots, vec(A_m)] \in \mathbb{R}^{n^2\times m},$$

and

$$\mathcal{A}(X) = A\,vec(X) \quad \text{and} \quad \mathcal{A}^T v = mat(A^T v),$$

where *mat* is the "inverse" operator to *vec* (i.e., $mat(vec(A_i)) = A_i \in S\mathbb{R}^{n\times n}$).

Finally, given a symmetric matrix G, let $G \odot G$ denote the operator from $S\mathbb{R}^{n\times n}$ to itself given by

$$(G \odot G)U = GUG.$$

The notation $U \otimes V$ indicates the standard Kronecker product of U and V.

2 The dual barrier algorithm

Let us consider the dual barrier problem parametrized by $\mu > 0$ (see [21,8])

$$\begin{array}{l} \max\ b^T y - \mu \ln(det(S)), \\ \text{s.t.}\ \ \mathcal{A}^T y + S = C. \end{array}$$

Let $X = \mu S^{-1} \succ 0$, then the first-order optimality conditions for this problem are given by:

$$F_\mu(X, y, S) = \begin{pmatrix} \mathcal{A}^T y + S - C \\ \mathcal{A}(X) - b \\ X - \mu S^{-1} \end{pmatrix} = 0. \tag{3}$$

We adopt the dual-path following method described in [21,8] that we will briefly describe below. Chosen a strictly dual feasible pair (y, S) and a scalar $\mu > 0$, damped Newton steps for the problem $F_\mu(X, y, S) = 0$ are made, maintaining S positive definite. Let (X_ℓ, y_ℓ, S_ℓ) be the current primal-dual iterate, then the Newton step $(\Delta X, \Delta y, \Delta S)$ is the solution of the following linear system

$$\begin{bmatrix} 0 & \mathcal{A}^T & I \\ \mathcal{A} & 0 & 0 \\ I & 0 & \mu(S_\ell^{-1} \odot S_\ell^{-1}) \end{bmatrix} \begin{bmatrix} \Delta X \\ \Delta y \\ \Delta S \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ X_\ell - \mu S_\ell^{-1} \end{bmatrix}. \tag{4}$$

Computing ΔX from the third equation in (4) (and applying earlier introduced notation $(S_\ell^{-1} \odot S_\ell^{-1})\Delta S = S_\ell^{-1}\Delta S\, S_\ell^{-1}$) gives

$$\Delta X = -\mu(S_\ell^{-1}\Delta S\, S_\ell^{-1}) - (X_\ell - \mu S_\ell^{-1}) \tag{5}$$

and letting $\Delta\tilde{S} = (S_\ell^{-1} \odot S_\ell^{-1})\Delta S$, we get the linear system in the augmented form

$$\begin{bmatrix} S_\ell \odot S_\ell & \mathcal{A}^T \\ \mathcal{A} & 0 \end{bmatrix} \begin{bmatrix} \Delta\tilde{S} \\ \Delta y \end{bmatrix} = - \begin{bmatrix} 0 \\ \frac{1}{\mu}\mathcal{A}(X_\ell - \mu S_\ell^{-1}) \end{bmatrix}. \tag{6}$$

The Schur complement form of the system can be obtained computing ΔS from the first equation in (4)

$$\Delta S = -\mathcal{A}^T \Delta y, \tag{7}$$

substituting it in (5) and getting

$$\Delta X = \mu(S_\ell^{-1}(\mathcal{A}^T\Delta y)S_\ell^{-1}) - (X_\ell - \mu S_\ell^{-1}). \tag{8}$$

Finally from the second equation in (4) we obtain:

$$M_\ell \Delta y = \frac{1}{\mu}\mathcal{A}(X_\ell - \mu S_\ell^{-1}). \tag{9}$$

where M_ℓ is the Schur complement matrix given by

$$M_\ell := A(S_\ell^{-1} \otimes S_\ell^{-1})A^T \in S\mathbb{R}^{m\times m}. \tag{10}$$

We note that the matrix M_ℓ is symmetric and positive definite, its entries are given by

$$(M_\ell)_{i,j} = (A_i S_\ell^{-1}) \bullet (S_\ell^{-1} A_j)$$

and it is generally dense.

Assuming that the initial guess is primal-dual feasible, primal-dual feasibility is maintained at each Newton iteration. Therefore one can substitute $\mathcal{A}(X_\ell)$ with b in the right hand side of (9) and solve the linear system

$$M_\ell \Delta y = \frac{1}{\mu} b - \mathcal{A}(S_\ell^{-1}). \tag{11}$$

This substitution allows to avoid using the primal variable X_ℓ explicitly.

The subsequent damped iterates are given by $y_{\ell+1} = y_\ell + \alpha\Delta y$ and $S_{\ell+1} = S_\ell + \alpha\Delta S$ with α such that $S_{\ell+1} \succ 0$. Given $S_\ell \succ 0$, we define

$$X_{\ell+1} = \underset{X}{\operatorname{argmin}} \left\{ \left\| \frac{S_\ell^{1/2}\, X\, S_\ell^{1/2}}{\mu} - I \right\|_F : \mathcal{A}(X) = b \right\} \tag{12}$$

and notice that $X_{\ell+1}$ has the following form:

$$X_{\ell+1} = \mu(S_\ell^{-1} + S_\ell^{-1}(\mathcal{A}^T\Delta y)S_\ell^{-1}), \tag{13}$$

see [8, Section 5.8]. From (8) we have $X_{\ell+1} = X_\ell + \Delta X$. The adopted *centrality measure* is

$$\delta(S_\ell, \mu) := \left\| \frac{S_\ell^{1/2}\, X_{\ell+1}\, S_\ell^{1/2}}{\mu} - I \right\|_F = \left\| S_\ell^{-1/2}\, \Delta S S_\ell^{-1/2} \right\|_F, \tag{14}$$

where in the last equality we have used (7) and (13). This measure is used in the proximity stopping criterion since the damped Newton process is carried out until

$$\delta(S_\ell, \mu) \le \tau$$

with $\tau \in (0, 1)$. Then the scalar μ is reduced and a new nonlinear system is solved.

3 The inexact dual-logarithmic barrier algorithm

We focus on sparse large dimension problems of the form (1)-(2) and propose an inexact version of the dual-logarithmic barrier algorithm for its solution. We therefore assume that the dual variable S is sparse, while the variable X may be dense and that the memory storage of the Schur complement matrix M_ℓ is prohibitive.

We fix the value of μ and use an Inexact Newton method. The step is made in a direction which is an approximate solution of the Schur complement formulation (11) computed in a matrix-free regime by using a Krylov method. The method is iterated until the centrality measure $\delta(S_\ell, \mu)$ drops below a prescribed tolerance. The barrier term μ is then reduced to force convergence.

Note that one might consider solving the augmented system (6) since both the application of the coefficient matrix and the computation of ΔS manage to avoid the inverse of S_ℓ hence are expected to be inexpensive. However, this formulation has two drawbacks. Firstly, $\Delta\tilde{S} = (S_\ell^{-1} \odot S_\ell^{-1})\Delta S$ is very likely to be dense despite ΔS being sparse. Secondly, solving linear system (11) inexactly causes ΔS to lose symmetry.

The overall Long-Step Inexact Interior-Point procedure is described in Algorithm 1. Clearly, the main step is the inexact solution of (3) at Line 4 whose steps are described in detail in Algorithm 2.

Algorithm 1 The Long-Step Inexact Dual-Logarithmic barrier algorithm

input: Dual feasible pair (y, S) (S positive definite), $\mu > 0$, $\epsilon > 0$, $\sigma \in (0, 1)$.
output: Approximation of the Primal-Dual solution (X, y, S).
1: $y_0 \leftarrow y$, $S_0 \leftarrow S$ and $\mu_0 \leftarrow \mu$.
2: **for** $k = 0, 1, \ldots$ **do**
3: Choose $\tau_k \in (0, 1)$.
4: $(y_{k+1}, S_{k+1}, \Delta S) \leftarrow$ Inexact_Newton$(y_k, S_k, \mu_k, \tau_k)$
5: **if** $\mu_k \leq \epsilon$ **then**
6: $X_{k+1} = \mu_k(S_k^{-1} - S_k^{-1}\,\Delta S\,S_k^{-1})$
7: **return** $X \leftarrow X_{k+1}, y \leftarrow y_{k+1}$ and $S \leftarrow S_{k+1}$.
8: **end if**
9: $\mu_{k+1} \leftarrow \sigma\mu_k$.
10: **end for**

Some comments on the algorithms are in order. First, note that in Algorithm 1 an initial primal variable X is not required. Also, if the dual vector solution is not needed, one can avoid to update the vector y_ℓ in Algorithm 2 explicitly and deal with Δy only. Second, the main task in Algorithm 2 is the computation of the step Δy at Lines 5-6. The computed step Δy satisfies

$$M_\ell \Delta y = \frac{1}{\mu} b - \mathcal{A}(S_\ell^{-1}) + r_\ell, \tag{17}$$

where the residual vector r_ℓ satisfies inequality (16). Hence it is an approximate solution of (11) and the relative residual norm is bounded by the forcing term $\eta_\ell > 0$. Third, the backtracking in the while loop at Lines 8-10 of Algorithm

Algorithm 2 The long-step Inexact-Newton algorithm

input: Dual feasible pair (y, S) (S positive definite), $\mu > 0$, $\tau \in (0, 1)$.
output: New dual feasible pair (y, S) and the last computed dual Newton step ΔS.

1: **procedure** INEXACT_NEWTON(y, S, μ, τ)
2: $S_0 \leftarrow S, y_0 \leftarrow y, \ell \leftarrow 0$ and $is\_centered \leftarrow 0$.
3: **while** $is\_centered = 0$ **do**
4: Choose $\eta_\ell \in (0, 1)$.
5: Compute an (inexact) step Δy such that

$$r_\ell = M_\ell \Delta y - \left(\frac{1}{\mu} b - \mathcal{A}(S_\ell^{-1})\right) \tag{15}$$

6: satisfies

$$\|r_\ell\| \leq \eta_\ell \left\| \frac{1}{\mu} b - \mathcal{A}(S_\ell^{-1}) \right\| \tag{16}$$

7: $\Delta S \leftarrow -\mathcal{A}^T(\Delta y)$ and $\alpha \leftarrow 1$.
8: **while** $S_\ell + \alpha \Delta S$ is not positive definite **do**
9: $\alpha = \alpha/2$.
10: **end while**
11: $y_{\ell+1} \leftarrow y_\ell + \alpha \Delta y$ and $S_{\ell+1} \leftarrow S_\ell + \alpha \Delta S$.
12: $\delta(S_\ell, \mu) \leftarrow \left\| S_\ell^{-1/2} \Delta S S_\ell^{-1/2} \right\|_F$.
13: **if** $\delta(S_\ell, \mu) \leq \tau$ **then**
14: $is\_centered \leftarrow 1$
15: **else**
16: $\ell \leftarrow \ell + 1$.
17: **end if**
18: **end while**
19: **return** $y \leftarrow y_\ell$, $S \leftarrow S_\ell$ and ΔS.
20: **end procedure**

2 preserves positive definiteness of S_ℓ, for each ℓ. Then, at each iteration k of Algorithm 1, matrix S_{k+1} computed at Line 4 is positive definite.

Moreover, in the inexact framework, we approximate the centrality measure in (14) with the following "inexact" measure

$$\delta(S_\ell, \mu) := \left\| S_\ell^{-1/2} \, \Delta S S_\ell^{-1/2} \right\|_F = \left\| \frac{S_\ell^{1/2} \, X_{\ell+1} \, S_\ell^{1/2}}{\mu} - I \right\|_F, \tag{18}$$

where ΔS is computed at Line 7 of Algorithm 2 and $X_{\ell+1}$ is defined by

$$X_{\ell+1} = \operatorname*{argmin}_X \left\{ \left\| \frac{S_\ell^{1/2} \, X \, S_\ell^{1/2}}{\mu} - I \right\|_F : \mathcal{A}(X) = b + \mu r_\ell \right\}. \tag{19}$$

Although the algebraic formulae in (14) and (18) are identical, their meaning is different because $X_{\ell+1}$ in the latter is allowed to violate the primal feasibility. The following result demonstrates that equality in (18) holds and the inexact computation of Δy in (15) yields a loss of primal feasibility.

Proposition 1 *Let $X_{\ell+1}$ be defined in (19). Then for each $\ell > 0$*

$$X_{\ell+1} = \mu(S_\ell^{-1} - S_\ell^{-1} \, \Delta S \, S_\ell^{-1}) \tag{20}$$

and

$$\left\| S_\ell^{-1/2} \, \Delta S S_\ell^{-1/2} \right\|_F = \left\| \frac{S_\ell^{1/2} \, X_{\ell+1} \, S_\ell^{1/2}}{\mu} - I \right\|_F . \tag{21}$$

Moreover, $X_{\ell+1}$ satisfies

$$\mathcal{A}(X_{\ell+1}) = b + \mu \, r_\ell .$$

Proof Equation (20) can be derived following [8, Section 5.8] and writing the first-order optimality conditions for the minimization problem in (19). Equality (21) easily follows from (20). Finally, we have

$$\begin{aligned}
\mathcal{A}(X_{\ell+1}) &= \mathcal{A}(\mu(S_\ell^{-1} - S_\ell^{-1} \, \Delta S \, S_\ell^{-1})) \\
&= \mu \mathcal{A}(S_\ell^{-1}) + \mu \mathcal{A}(S_\ell^{-1} \odot S_\ell^{-1}) \mathcal{A}^T \Delta y \\
&= \mu \mathcal{A}(S_\ell^{-1}) + \mu M_\ell \Delta y \\
&= \mu \mathcal{A}(S_\ell^{-1}) + \mu \left(\frac{1}{\mu} b - \mathcal{A}(S_\ell^{-1}) + r_\ell . \right) \\
&= b + \mu \, r_\ell .
\end{aligned}$$

□

Then, letting ΔX as in (5) we obtain

$$X_{\ell+1} = X_\ell + \Delta X. \tag{22}$$

Interestingly, dual feasibility is preserved even if inexact steps are used. Assuming the pair (y_ℓ, S_ℓ) satisfies $\mathcal{A}^T y_\ell + S_\ell = C$, then dual feasibility of $(S_\ell + \alpha \Delta S, y_\ell + \alpha \Delta y)$ follows from the feasibility of (S_ℓ, y_ℓ) and the definition of ΔS in Line 7 of Algorithm 2:

$$\mathcal{A}^T(y_\ell + \alpha \Delta y) + S_\ell + \alpha \Delta S = (\mathcal{A}^T y_\ell + S_\ell) + \alpha(\mathcal{A}^T \Delta y + \Delta S) = C + 0.$$

In order to distinguish the "inexact" measure (18) from the "exact" one in (14), we use the subscript ex for (14). Analogously, we denote with $(\Delta X_{ex}, \Delta y_{ex}, \Delta S_{ex})$ the "exact" solution of (4) which corresponds to having $r_\ell = 0$ in (15). Then,

$$\delta_{ex}(S_\ell, \mu) = \left\| \frac{S_\ell^{1/2} \, X_{\ell+1,ex} \, S_\ell^{1/2}}{\mu} - I \right\|_F$$

and

$$X_{\ell+1,ex} = \mu(S_\ell^{-1} - S_\ell^{-1} \, \Delta S_{ex} \, S_\ell^{-1}) = X_\ell + \Delta X_{ex}. \tag{23}$$

The relationship between the two centrality measures follows.

Lemma 1 *Let $S_\ell \succ 0$. Then*

$$|\delta_{ex}(S_\ell, \mu) - \delta(S_\ell, \mu)| \leq \|A^\dagger\|_2 \|S_\ell\|_2 \|r_\ell\|$$

Proof Combining (22) and (23) we get $X_{\ell+1,ex} = X_{\ell+1} + \Delta X_{ex} - \Delta X$ and then

$$\begin{aligned}\delta_{ex}(S_\ell,\mu) &= \left\| \frac{S_\ell^{1/2}\, X_{\ell+1,ex}\, S_\ell^{1/2}}{\mu} - I \right\|_F \\ &= \left\| \frac{S_\ell^{1/2}\,(X_{\ell+1} - \Delta X + \Delta X_{ex})\, S_\ell^{1/2}}{\mu} - I \right\|_F \\ &\le \delta(S_\ell,\mu) + \left\| \frac{S_\ell^{1/2}\,(\Delta X - \Delta X_{ex})\, S_\ell^{1/2}}{\mu} \right\|_F . \end{aligned} \tag{24}$$

Moreover, (20) and (23) yield

$$\begin{aligned}\Delta X - \Delta X_{ex} &= X_{\ell+1} - X_{\ell+1,ex} \\ &= \mu(S_\ell^{-1}(\Delta S_{ex} - \Delta S)S_\ell^{-1}) \\ &= \mu(S_\ell^{-1}(\mathcal{A}^T(\Delta y - \Delta y_{ex}))S_\ell^{-1}).\end{aligned}$$

Since $M_\ell \Delta y = M_\ell \Delta y_{ex} + r_\ell$ we get

$$\begin{aligned}\left\| \frac{S_\ell^{1/2}\,(\Delta X - \Delta X_{ex})\, S_\ell^{1/2}}{\mu} \right\|_F &= \left\| S_\ell^{-1/2}\,(\mathcal{A}^T(M_\ell^{-1} r_\ell))\, S_\ell^{-1/2} \right\|_F \\ &= \left\| (S_\ell^{-1/2} \otimes S_\ell^{-1/2}) A^T (A(S_\ell^{-1} \otimes S_\ell^{-1}) A^T)^{-1} r_\ell) \right\|_2 .\end{aligned}$$

Let $G := A(S_\ell^{-1/2} \otimes S_\ell^{-1/2})$, then

$$\left\| \frac{S_\ell^{1/2}\,(\Delta X - \Delta X_{ex})\, S_\ell^{1/2}}{\mu} \right\|_F = \left\| G^T (G G^T)^{-1} r_\ell \right\| = \left\| G^\dagger r_\ell \right\| .$$

We observe that the vector $\hat{q} := G^\dagger r_\ell \in \mathbb{R}^n$ is the (unique) minimum length solution of the least-squares problem $\min_{q\in\mathbb{R}^n} \|Gq - r_\ell\|$. Let $A = U\Sigma V^T$ be the SVD of A. Then letting $z = (S_\ell^{-1/2} \otimes S_\ell^{-1/2})\hat{q}$,

$$\|G\hat{q} - r_\ell\| = \|Az - r_\ell\| = \|U\Sigma V^T z - r_\ell\| = \|\Sigma g - U^T r_\ell\|,$$

where $g = V^T z$. Therefore $g = \Sigma^\dagger U^T r_\ell$, $z = V\Sigma^\dagger U^T r_\ell$ and finally

$$\|\hat{q}\| = \|(S_\ell^{1/2} \otimes S_\ell^{1/2}) z\| \le \|S_\ell\|_2 \|A^\dagger\|_2 \|r_\ell\|.$$

We therefore obtain by (24)

$$\delta_{ex}(S_\ell,\mu) \le \delta(S_\ell,\mu) + \|A^\dagger\|_2 \|S_\ell\|_2 \|r_\ell\|.$$

On the other hand, writing $X_{\ell+1} = X_{\ell+1,ex} + \Delta X - \Delta X_{ex}$ and following the steps above, we obtain

$$\delta(S_\ell,\mu) \le \delta_{ex}(S_\ell,\mu) + \|A^\dagger\|_2 \|S_\ell\|_2 \|r_\ell\|,$$

which completes the proof. □

Now, we bound the duality gap in terms of the centrality measure and the residual vector.

Lemma 2 *At each ℓ-th iteration of Algorithm 2 the following inequality holds*

$$\mu(1-\delta(S_\ell,\mu)/\sqrt{n}+(r_\ell^T y_\ell)/n) \leq \frac{C \bullet X_{\ell+1} - b^T y_\ell}{n} \leq \mu(1+\delta(S_\ell,\mu)/\sqrt{n}+(r_\ell^T y_\ell)/n)$$

Proof Note that using Proposition 1 we have $b = \mathcal{A}(X_{\ell+1}) - \mu r_\ell$, hence, following the proof of Lemma 5.8 in [8] we get

$$\begin{aligned} C \bullet X_{\ell+1} - b^T y_\ell &= C \bullet X_{\ell+1} - (\mathcal{A}(X_{\ell+1}) - \mu r_\ell)^T y_\ell \\ &= (C - \mathcal{A}^T y_\ell) \bullet X_{\ell+1} + \mu r_\ell^T y_\ell. \end{aligned}$$

Using the dual feasibility of (y_ℓ, S_ℓ) we get

$$C \bullet X_{\ell+1} - b^T y_\ell = S_\ell \bullet X_{\ell+1} + \mu r_\ell^T y_\ell. \tag{25}$$

Using the Cauchy-Schwartz inequality we have

$$\delta(S_\ell,\mu)\sqrt{n} = \left\| \frac{S_\ell^{1/2} X_{\ell+1} S_\ell^{1/2}}{\mu} - I \right\|_F \|I\|_F \geq \left| \frac{S_\ell \bullet X_{\ell+1}}{\mu} - n \right|$$

that implies

$$n - \delta(S_\ell,\mu)\sqrt{n} \leq \frac{S_\ell \bullet X_{\ell+1}}{\mu} \leq n + \delta(S_\ell,\mu)\sqrt{n}. \tag{26}$$

We get the thesis combing the above equation with (25). □

4 Convergence analysis of the Inexact Newton inner procedure

In this section we will look into convergence properties of Algorithm 2. We start from observing that once the stopping criterion at Line 13 of Algorithm 2 has been satisfied, say at iteration $\bar{\ell}$, the corresponding matrix $X_{\bar{\ell}+1}$ defined in (20) is positive definite.

Lemma 3 *Let $S_\ell \succ 0$ and $\delta(S_\ell,\mu) < 1$. Then $X_{\ell+1} \succ 0$.*

Proof The proof follows by a straightforward modification of the proof of Lemma 5.3 in [8]. □

Moreover, the backtracking process in the while loop at Lines 8-10 is well-defined and if S_ℓ is sufficiently well-centered, then $\alpha = 1$ is taken. This is proved in the next two Lemmas.

Lemma 4 *Assume $S_\ell \succ 0$. Then, there exists $\bar{\alpha} > 0$ such that $S_\ell + \alpha \Delta S \succ 0$ for any α in $(0, \bar{\alpha})$*

Proof Assume ΔS is not positive definite, otherwise $S_\ell + \alpha\Delta S \succ 0$ for any $\alpha \geq 0$. We have

$$S_\ell + \alpha\Delta S = S_\ell^{1/2}(I + \alpha S_\ell^{-1/2}\Delta S S_\ell^{-1/2})S_\ell^{1/2}.$$

Then, as $S_\ell^{1/2} \succ 0$ by hypothesis, it follows that $S_\ell + \alpha\Delta S \succ 0$ whenever α is sufficiently small. In particular, the thesis holds with $\bar{\alpha} = 1/\|S_\ell^{-1/2}\Delta S S_\ell^{-1/2}\|_F$. □

Lemma 5 *Let $S_\ell \succ 0$ and $\delta(S_\ell, \mu_k) < 1$. Then $S_\ell + \Delta S \succ 0$.*

Proof The proof follows by a straightforward modification of the proof of Lemma 5.4 in [8]. □

We observe that the sequence $\{(y_\ell, S_\ell)\}$, computed by Algorithm 2 corresponds to the sequence generated by an Inexact Newton method applied to the reduced nonlinear system:

$$\tilde{F}_\mu(y, S) = \begin{pmatrix} \mathcal{A}^T y + S - C \\ \mu\mathcal{A}(S^{-1}) - b \end{pmatrix} = 0. \tag{27}$$

In fact, the Newton system is given by:

$$\begin{bmatrix} \mathcal{A}^T & I \\ 0 & \mu\mathcal{A}(S^{-1} \odot S^{-1}) \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta S \end{bmatrix} = - \begin{bmatrix} 0 \\ b - \mu\mathcal{A}(S_\ell^{-1}) \end{bmatrix}.$$

Then, the vector $(\Delta y, \Delta S)$ computed at Lines 5 and 7 is an Inexact Newton step for (27) as it satisfies

$$\begin{bmatrix} \mathcal{A}^T & I \\ 0 & \mu\mathcal{A}(S^{-1} \odot S^{-1}) \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta S \end{bmatrix} = - \begin{bmatrix} 0 \\ b - \mu\mathcal{A}(S_\ell^{-1}) \end{bmatrix} - \begin{bmatrix} 0 \\ \mu r_\ell \end{bmatrix}. \tag{28}$$

Moreover, by (16) the residual vector satisfies the following inequality

$$\left\| \begin{bmatrix} 0 \\ \mu r_\ell \end{bmatrix} \right\| \leq \mu\eta_\ell \left\| \frac{1}{\mu} b - \mathcal{A}(S_\ell^{-1}) \right\| \leq \eta_\ell \left\| \tilde{F}_\mu(y_\ell, S_\ell) \right\|. \tag{29}$$

Then, if we equip Algorithm 2 with a line-search along the direction $(\Delta y, \Delta S)$ we obtain an Inexact Newton method in the framework of Algorithm INB in [9]. Therefore, given $t \in (0, 1)$, let us substitute the loop at Lines 8-10 of Algorithm 2 with the steps described in Algorithm 3.

Algorithm 3 Backtracking strategy to enforce a decrease of $\|\tilde{F}_\mu\|$ and positive definiteness of S

1: **while** $S_\ell + \alpha\Delta S$ is not positive definite or
2: $\|\tilde{F}_\mu(y_\ell + \alpha\Delta y, S_\ell + \alpha\Delta S)\| > (1 - t(1 - \eta_\ell))\|\tilde{F}_\mu(y_\ell, S_\ell)\|$ **do**
3: $\alpha = \alpha/2$ and $\eta_\ell = (1 - (1 - \eta_\ell)/2)$.
4: **end while**

It is worth pointing out that when the inexact Newton method is applied to find an approximate solution of (27) the value of barrier term μ remains constant. This is independent of which mechanism is used to choose the step-size (either Lines 8-10 of Algorithm 2 or the backtracking strategy suggested in Algorithm 3). Moreover, letting $\Delta\tilde{S} = (S_\ell^{-1} \odot S_\ell^{-1})\Delta S$ the linear system (28) may be rewritten as

$$\begin{bmatrix} (S_\ell \odot S_\ell)\ \mathcal{A}^T \\ \mathcal{A} \qquad 0 \end{bmatrix} \begin{bmatrix} \Delta\tilde{S} \\ \Delta y \end{bmatrix} = - \begin{bmatrix} 0 \\ r_\ell + b/\mu - \mathcal{A}(S_\ell^{-1}) \end{bmatrix}.$$

Its coefficient matrix is nonsingular provided that $S_\ell \succ 0$ and this is indeed the case since μ is a fixed positive constant in this context. Therefore $\tilde{F}'_\mu(y_\ell, S_\ell)$ is nonsingular for any $\ell \geq 0$.

Assume that the sequences $\{\|S_\ell\|\}_{\ell=1}^{\infty}$, $\{\|y_\ell\|\}_{\ell=1}^{\infty}$ are bounded. Then, the sequence $\{(y_\ell, S_\ell)\}$ has at least one accumulation point and from Theorem 6.1 in [9] it follows that if there exists a limit point (y_*, S_*) of $\{y_\ell, S_\ell\}$ such that $\tilde{F}'_\mu(y_*, S_*)$ is nonsingular, then $\tilde{F}_\mu(y_*, S_*) = 0$ and $\{y_\ell, S_\ell\} \to (y_*, S_*)$, whenever ℓ goes to infinity. As a consequence we have

$$\lim_{\ell\to\infty} \|\tilde{F}_\mu(y_\ell, S_\ell)\| = 0 \tag{30}$$

as the sequence $\{\|\tilde{F}_\mu(y_\ell, S_\ell)\|\}$ is monotonically decreasing and bounded from below. This together with (29) implies that

$$\lim_{\ell\to\infty} \|r_\ell\| = 0. \tag{31}$$

We now show that the stopping criterion at Line 13 of Algorithm 2 is satisfied after a finite number of inner iterations. First, from (20), and (15) it follows

$$\begin{aligned} \|X_{\ell+1} - \mu S_\ell^{-1}\|_F &= \|\mu(S_\ell^{-1} \otimes S_\ell^{-1})A^T \Delta y\| \\ &= \|(S_\ell^{-1} \otimes S_\ell^{-1})A^T M_\ell^{-1}(r_\ell + \frac{b}{\mu} - \mathcal{A}(S_\ell^{-1}))\| \\ &= \|(S_\ell^{-1/2} \otimes S_\ell^{-1/2})G_\ell^T (G_\ell G_\ell^T)^{-1}(r_\ell + \frac{b}{\mu} - \mathcal{A}(S_\ell^{-1}))\| \\ &= \|(S_\ell^{-1/2} \otimes S_\ell^{-1/2})G_\ell^\dagger (r_\ell + \frac{b}{\mu} - \mathcal{A}(S_\ell^{-1}))\| \end{aligned}$$

with $G_\ell = A(S_\ell^{-1/2} \otimes S_\ell^{-1/2})$. Then, proceeding as in the proof of Lemma 1, i.e. letting $A = U\Sigma V^T$ be the SVD of A, $\hat{q} = G_\ell^\dagger(r_\ell + \frac{b}{\mu} - \mathcal{A}(S_\ell^{-1}))$ and $z = (S_\ell^{-1/2} \otimes S_\ell^{-1/2})\hat{q}$, we get

$$z = V\Sigma^\dagger U^T(r_\ell + \frac{b}{\mu} - \mathcal{A}(S_\ell^{-1}))$$

and

$$\begin{aligned}\|X_{\ell+1} - \mu S_\ell^{-1}\|_F = \|z\| &\leq \|A^\dagger\|_2 \left(\|r_\ell\| + \left\|b/\mu - \mathcal{A}(S_\ell^{-1})\right\|\right) \\ &\leq \|A^\dagger\|_2 \left(\|r_\ell\| + \mu\|\tilde{F}_\mu(y_\ell, S_\ell)\|\right).\end{aligned}$$

This yields $\|X_{\ell+1} - \mu S_\ell^{-1}\|_F \to 0$ using (30) and (31). Then, as

$$\delta(S_\ell, \mu) \leq \frac{1}{\mu}\|S_\ell\|_F \, \|X_{\ell+1} - \mu S_\ell^{-1}\|_F$$

and $\|S_\ell\|_F$ is bounded by assumption we can conclude that $\delta(S_\ell, \mu) \to 0$.

Finally, we have that primal feasibility is recovered since

$$\begin{aligned}\|\mathcal{A}(X_{\ell+1}) - b\| &\leq \|\mathcal{A}(X_{\ell+1} - \mu S_\ell^{-1})\| + \|b - \mu\mathcal{A}(S_\ell^{-1})\| \\ &\leq \|A\|_2 \|X_{\ell+1} - \mu S_\ell^{-1}\|_F + \|\tilde{F}_\mu(y_\ell, S_\ell)\|\end{aligned}$$

and therefore $\|\mathcal{A}(X_{\ell+1}) - b\| \to 0$ whenever ℓ goes to ∞.

Then, in order to compute accurate primal solution, in the last iteration of Algorithm 1 a small tolerance τ_k is set to force the Inexact Newton method to iterate till convergence.

5 A short-step version of the inexact dual-logarithmic barrier method

In this section we introduce a short-step version of Algorithms 1 and 2. This corresponds to use a more conservative update $\mu_{k+1} = \sigma\mu_k$ with $\sigma = (1-\theta)$ for some small θ, to perform only two Inexact Newton iterations for each μ-value and to consider an initial pair (y, S) sufficiently well centered with respect to the initial μ. In particular, a sequence is generated such that if the initial S is sufficiently close to the central path, i.e. such that $\delta(S, \mu) \leq 1/2$, then after two inexact Newton steps the new approximation serves as well-centered starting iterate for the sub-sequent outer iteration. In this section we will use two indices to specify the outer k-th iteration and the three inner iterations $\ell = 0, 1, 2$.

In order to establish such convergence properties we need to modify the accuracy requirement in the solution of the linear systems. Indeed, the norm of the residual is controlled by the value of the centrality measure $\delta(S_{k,\ell}, \mu_k)$ instead of $\|b - \mu_k\mathcal{A}(S_{k,\ell}^{-1})\|$ as in the long-step version. We state the short-step version of the inexact dual-logarithmic barrier method in Algorithm 4.

Note that since $r_{0,0} = 0$, then $X_{0,1}$ is primal feasible. Moreover, Lemmas 3–5 hold also for the short-step version and for its "exact" counterpart, as the accuracy requirement in the solution of linear systems is never involved in their proofs. The following observations about the exact counterpart of our short-step algorithm are in order. First, the following result on the decrease of centrality measure holds.

Algorithm 4 The Short-Step Inexact Dual-Logarithmic barrier algorithm

input: Dual feasible S and $\mu > 0$ such that $\delta_{ex}(S,\mu) \leq \frac{1}{2}$, $\gamma, \beta > 0$, $\epsilon > 0, \theta \in (0,1)$.
output: Primal-Dual solution (X, S).

$S_{0,0} \leftarrow S$ and $\mu_0 \leftarrow \mu$.
while $n\mu_k > \epsilon$ **do**
  First Newton Step
  if $k = 0$ **then**
    Compute the solution Δy of

$$M_{k,0}\Delta y = \frac{1}{\mu_k} b - \mathcal{A}(S_{k,0}^{-1})$$

  else
    Compute an inexact step Δy such that

$$r_{k,0} = M_{k,0}\Delta y - \left(\frac{1}{\mu_k} b - \mathcal{A}(S_{k,0}^{-1})\right)$$

    satisfies

$$\|r_{k,0}\| \leq \gamma \frac{1}{\|A^\dagger\|_2 \|S_{k,0}\|_2}\, \delta^2(S_{k-1,1}, \mu_{k-1}) \tag{32}$$

  end if
  $\Delta S \leftarrow -(\mathcal{A}^T \Delta y)$.
  $S_{k,1} \leftarrow S_{k,0} + \Delta S$.
  $\delta(S_{k,0}, \mu_k) \leftarrow \|S_{k,0}^{-1/2} \Delta S S_{k,0}^{-1/2}\|_F$.
  Second Newton Step
  Compute an inexact step Δy such that

$$r_{k,1} = M_{k,1}\Delta y - \left(\frac{1}{\mu_k} b - \mathcal{A}(S_{k,1}^{-1})\right) \tag{33}$$

  satisfies

$$\|r_{k,1}\| \leq \beta \frac{1}{\|A^\dagger\|_2 \|S_{k,1}\|_2}\, \delta^2(S_{k,0}, \mu_k). \tag{34}$$

  $\Delta S \leftarrow -(\mathcal{A}^T \Delta y)$.
  $S_{k,2} \leftarrow S_{k,1} + \Delta S$.
  $\delta(S_{k,1}, \mu_k) \leftarrow \|S_{k,1}^{-1/2} \Delta S S_{k,1}^{-1/2}\|_F$.
  Update the iterates
  $S_{k+1,0} \leftarrow S_{k,2}$.
  $\mu_{k+1} \leftarrow (1-\theta)\mu_k$.
end while
return $X \leftarrow \mu_k(S_{k,1}^{-1} - S_{k,1}^{-1}\, \Delta S\, S_{k,1}^{-1})$ and $S \leftarrow S_{k+1,0}$.

Lemma 6 *[8, Lemma 5.5] Fixed $k \geq 0$, if $S_{k,\ell} \succ 0$ is dual feasible and $\delta_{ex}(S_{k,\ell}, \mu_k) < 1$, then*

$$\delta_{ex}(S_{k,\ell+1}, \mu_k) \leq \delta_{ex}^2(S_{k,\ell}, \mu_k),$$

with $\ell = 0, 1$.

Second, it is shown in [8] that starting from $S_{k,0}$ such that $\delta_{ex}(S_{k,0}, \mu_k) < \frac{1}{2}$, after one Newton step the obtained approximation $S_{k,1}$ can be used as the subsequent well-centered starting point $S_{k+1,0}$, i.e. such that $\delta_{ex}(S_{k+1,0}, \mu_{k+1}) \leq$

$\frac{1}{2}$ and the process is quadratically convergent. In the inexact case, we are going to prove that we need two inexact Newton iterations to get an analogous result. In other words we are going to show that the Inexact Newton process is two-step quadratically convergent in terms of the centrality measure. Obviously, as in the long step case, in the last outer iteration, we need to iterate the Inexact Newton method till convergence in order to recover primal feasibility. Therefore after termination of Algorithm 4, an extra outer iteration is performed: several steps of Inexact Newton method are made until the nonlinear residual is small.

Remark 1 In the following proofs we will make use of the centrality measure evaluated at $S_{k,2}$, i.e.

$$\delta(S_{k,2},\mu_k) = \left\| S_{k,2}^{-1/2}\, \Delta S S_{k,2}^{-1/2} \right\|_F, \tag{35}$$

where $\Delta S = -\mathcal{A}^T \Delta y$ and Δy is an approximate solution of the linear system:

$$M_{k,2}\Delta y = \frac{1}{\mu_k} b - \mathcal{A}(S_{k,2}^{-1})$$

with residual

$$\|r_{k,2}\| \le \gamma \frac{1}{\|A^\dagger\|_2 \|S_{k,2}\|_2}\, \delta^2(S_{k,1},\mu_k). \tag{36}$$

This quantity will be used to provide a bound for $\delta(S_{k+1,0},\mu_{k+1})$ that is computed at Line 12 of the subsequent $(k+1)$-th iteration in Algorithm 4. The computation of $\delta(S_{k,2},\mu_k)$ would require a third Newton step but in fact it is not needed in the algorithm.

Lemma 7 *Let $k \ge 0$, $S_{k,0} \succ 0$ be dual feasible and*

$$\max\{\delta(S_{k,1},\mu_k), \delta(S_{k,0},\mu_k)\} \le 1/2. \tag{37}$$

Moreover, let $\gamma > 0$, $\beta \in (0,2)$ and $\delta(S_{k,2},\mu_k)$ be given in (35).
Then, starting from $S_{k,0}$, the following inequality holds

$$\delta(S_{k,2},\mu_k) \le C(\gamma,\beta)\, \max\{\delta^2(S_{k,1},\mu_k), \delta^2(S_{k,0},\mu_k)\} \tag{38}$$

with

$$C(\gamma,\beta) = (1+\beta)^2 + \gamma. \tag{39}$$

Proof First, note that by Lemma 1 and (34) it follows

$$\begin{aligned} \delta_{ex}(S_{k,1},\mu_k) &\le \delta(S_{k,1},\mu_k) + \|A^\dagger\|_2 \|S_{k,1}\|_2 \|r_{k,1}\| \\ &\le \delta(S_{k,1},\mu_k) + \beta\delta^2(S_{k,0},\mu_k) \\ &\le 1/2 + \beta/4 < 1. \end{aligned} \tag{40}$$

Then, using again Lemma 1, Lemma 6 and (36) we get

$$\begin{aligned} \delta(S_{k,2},\mu_k) &\le \delta_{ex}(S_{k,2},\mu_k) + \|A^\dagger\|_2 \|S_{k,2}\|_2 \|r_{k,2}\| \\ &\le \delta_{ex}^2(S_{k,1},\mu_k) + \|A^\dagger\|_2 \|S_{k,2}\|_2 \|r_{k,2}\| \\ &\le \delta_{ex}^2(S_{k,1},\mu_k) + \gamma\delta^2(S_{k,1},\mu_k) \end{aligned}$$

and (40) and (37) yield

$$\delta_{ex}^2(S_{k,1},\mu_k) \le (1+\beta)^2 \max\{\delta^2(S_{k,1},\mu_k),\delta^2(S_{k,0},\mu_k)\}.$$

Therefore we have

$$\begin{aligned}\delta(S_{k,2},\mu_k) &\le (1+\beta)^2 \max\{\delta^2(S_{k,1},\mu_k),\delta^2(S_{k,0},\mu_k)\} + \gamma\delta^2(S_{k,1},\mu_k)\\ &\le C(\gamma,\beta) \max\{\delta^2(S_{k,1},\mu_k),\delta^2(S_{k,0},\mu_k)\}.\end{aligned}$$

with $C(\gamma,\beta)$ given in (39). □

The next step consists in bounding the centrality measure at the beginning of the outer iteration $k+1$ in terms of the centrality measure at the end of the previous iteration k.

Lemma 8 *Same assumptions as in Lemma 7. Then if $\mu_{k+1} = (1-\theta)\mu_k$ for some $\theta \in (0,1)$, then*

$$\delta(S_{k+1,0},\mu_{k+1}) \le \frac{1}{1-\theta}\left(\delta(S_{k,2},\mu_k) + \theta\sqrt{n} + \gamma(2-\theta)\right).$$

Proof Proceeding as in Lemma 1 and recalling that $S_{k+1,0} = S_{k,2}$ (Line 20 of Algorithm 4) we have

$$\begin{aligned}\delta(S_{k+1,0},\mu_{k+1}) &\le \delta_{ex}(S_{k+1,0},\mu_{k+1}) + \|S_{k+1,0}\|_2\|A^\dagger\|_2\|r_{k+1,0}\|\\ &= \left\|\frac{S_{k,2}^{1/2}\,\bar{X}_{k+1,1}^{ex}\,S_{k,2}^{1/2}}{\mu_{k+1}} - I\right\|_F + \|S_{k+1,0}\|_2\|A^\dagger\|_2\|r_{k+1,0}\| \quad (41)\end{aligned}$$

where

$$\bar{X}_{k+1,1}^{ex} = \operatorname*{argmin}_X \left\{\left\|\frac{S_{k+1,0}^{1/2}\,X\,S_{k+1,0}^{1/2}}{\mu_{k+1}} - I\right\|_F : \mathcal{A}(X) = b\right\}.$$

Using the minimization property of $\bar{X}_{k+1,1}^{ex}$ and the form of μ_{k+1} we obtain

$$\begin{aligned}\left\|\frac{S_{k,2}^{1/2}\,\bar{X}_{k+1,1}^{ex}\,S_{k,2}^{1/2}}{\mu_{k+1}} - I\right\|_F &\le \frac{1}{1-\theta}\left\|\frac{S_{k,2}^{1/2}\,X_{k+1,1}^{ex}\,S_{k,2}^{1/2}}{\mu_k} - (1-\theta)I\right\|_F\\ &\le \frac{1}{1-\theta}(\delta_{ex}(S_{k,2},\mu_k) + \theta\|I\|_F) \quad (42)\end{aligned}$$

where

$$X_{k+1,1}^{ex} = \operatorname*{argmin}_X \left\{\left\|\frac{S_{k+1,0}^{1/2}\,X\,S_{k+1,0}^{1/2}}{\mu_k} - I\right\|_F : \mathcal{A}(X) = b\right\}.$$

Combining (41), (42) and (32) we obtain

$$\delta(S_{k+1,0},\mu_{k+1}) \le \frac{1}{1-\theta}\left(\delta_{ex}(S_{k,2},\mu_k) + \theta\sqrt{n}\right) + \gamma.$$

where we used that by assumption (37) $\delta(S_{k,1},\mu_k)<1$. Using Lemma 1 and (36) we have

$$\begin{aligned}\delta_{ex}(S_{k,2},\mu_k) &\le \delta(S_{k,2},\mu_k)+\|S_{k,2}\|_2\|A^\dagger\|_2\|r_{k,2}\|\\ &\le \delta(S_{k,2},\mu_k)+\gamma.\end{aligned}$$

and we finally get the thesis. □

The next result establishes a bound on the proximity of $S_{k+1,0}$ to the central path.

Lemma 9 *Same assumptions as in Lemma 7. If*

$$\theta=\frac{2-C(\gamma,\beta)-8\gamma}{4\sqrt{n}-4\gamma+2} \tag{43}$$

with $C(\gamma,\beta)$ defined as in (39) and γ,β sufficiently small so that $\theta\in(0,1)$, then

$$\delta(S_{k+1,0},\mu_{k+1})\le\frac{1}{2}.$$

Proof Using Lemma 8 and Lemma 7 have

$$\begin{aligned}\delta(S_{k+1,0},\mu_{k+1}) &\le \frac{1}{1-\theta}\left(\delta(S_{k,2},\mu_k)+\theta\sqrt{n}+\gamma(2-\theta)\right)\\ &\le \frac{1}{1-\theta}\left(C(\gamma,\beta)\max\{\delta^2(S_{k,1},\mu_k),\delta^2(S_{k,0},\mu_k)\}+\theta\sqrt{n}+\gamma(2-\theta)\right)\end{aligned}$$

and (43) yields

$$\delta(S_{k+1,0},\mu_{k+1})\le\frac{1}{1-\theta}\left(C(\gamma,\beta)\frac{1}{4}+\theta\sqrt{n}+\gamma(2-\theta)\right)=\frac{1}{2}.$$

□

Now we show that at each iteration k, $S_{k,0}$ can be used as a well-centered starting point provided that the initial guess $S_{0,0}$ is well centered.

Lemma 10 *Let θ be given in (43) and $\mu_0>0$. Assume*

$$\delta_{ex}(S_{0,0},\mu_0)\le\frac{1}{2},$$

and $\gamma,\beta\in(0,1)$ in (39) are sufficiently small such that $\theta\in(0,1)$ and $4\gamma+\gamma^2+4\beta\le 4$. Then

$$\delta(S_{k+1,0},\mu_{k+1})\le\frac{1}{2},\ \ \textit{for any}\ \ k\ge 0.$$

Proof Let us consider the first outer iteration ($k = 0$). Then, as $r_{0,0} = 0$, it follows that $\delta(S_{0,0}, \mu_0) = \delta_{ex}(S_{0,0}, \mu_0)$, and by Lemma 1, we get

$$\delta(S_{0,1}, \mu_0) \leq \delta_{ex}(S_{0,1}, \mu_0) + \|A^\dagger\|_2 \|S_{0,1}\|_2 \|r_{0,1}\|.$$

Then, using Lemma 6, (34), and $\delta(S_{0,0}, \mu_0) \leq 1/2$ we get

$$\delta(S_{0,1}, \mu_0) \leq 1/2(1+\beta)\delta(S_{0,0}, \mu_0).$$

This implies $\delta(S_{0,1}, \mu_0) \leq \delta(S_{0,0}, \mu_0) \leq 1/2$ as $\beta < 1$ by hypothesis. Therefore, Lemma 9 ensures

$$\delta(S_{1,0}, \mu_1) \leq 1/2.$$

Let us consider $\delta(S_{1,1}, \mu_1)$. We are going to show that $\delta(S_{1,1}, \mu_1) \leq 1/2$. We have, using Lemma 6 and Lemma 1,

$$\begin{aligned}\delta(S_{1,1}, \mu_1) &\leq \delta_{ex}(S_{1,0}, \mu_1)^2 + \|A^\dagger\|_2 \|S_{1,1}\|_2 \|r_{1,1}\| \\ &\leq \left(\delta(S_{1,0}, \mu_1) + \|A^\dagger\|_2 \|S_{1,0}\|_2 \|r_{1,0}\|\right)^2 + \|A^\dagger\|_2 \|S_{1,1}\|_2 \|r_{1,1}\|.\end{aligned}$$

Then the stopping rules (32) and (34) and $\max\{\delta(S_{1,0}, \mu_1), \delta(S_{0,1}, \mu_0)\} \leq 1/2$ yield

$$\delta(S_{1,1}, \mu_1) \leq \left(\frac{1}{2} + \frac{\gamma}{4}\right)^2 + \frac{\beta}{4}.$$

Therefore, $\delta(S_{1,1}, \mu_1) \leq 1/2$ provided that $4\gamma + \gamma^2 + 4\beta \leq 4$. Then, Lemma 9 yields $\delta(S_{2,0}, \mu_2) \leq 1/2$. Proceeding in this way we can prove that $\delta(S_{k,1}, \mu_k) \leq 1/2$ for $k \geq 0$. □

Summing up, we have proved that, starting from a dual feasible point $S_{0,0}$ such that $\delta_{ex}(S_{0,0}, \mu_0) \leq 1/2$, at a generic iteration k performing two inexact Newton steps and reducing μ_k by a factor $(1-\theta)$, we get $\delta(S_{k+1,0}, \mu_{k+1}) \leq \frac{1}{2}$ at the subsequent iteration.

Theorem 1 *Let ϵ be an accuracy parameter, θ given in (43) and $\mu > 0$. Assume S is strictly dual feasible such that $\delta_{ex}(S, \mu) \leq 1/2$ and γ and β in (39) are sufficiently small such that $\theta \in (0,1)$ and $2\beta + \beta^2 + 9\gamma < 1/3$ and $4\gamma + \gamma^2 + 4\beta \leq 4$.*

Then,

(i) Algorithm 4 terminates after at most $\lceil 18\sqrt{n} \log \frac{n\mu}{\epsilon} \rceil$ inexact Newton steps.
(ii) Let $\bar{k}$ be the last iteration of Algorithm 4, then the following inequality holds:

$$S_{\bar{k},1} \bullet X_{\bar{k},2} \leq 3/2\epsilon$$

and

$$C \bullet X_{\bar{k},2} - b^T y_{\bar{k},1} \leq 3/2\epsilon + r_{\bar{k},1}^T y_{\bar{k},1}.$$

Proof We follow the lines of proof of Theorem 5.1 in [8]. At the end of each k-th iteration of Algorithm 4 $S_{k,2}$ is strictly feasible, $\mu_k = (1-\theta)^k \mu_0$ ($\mu_0 = \mu$ at Line 1) and the algorithm stops at iteration $\bar{k}$ where

$$n\mu_{\bar{k}} \leq \epsilon. \tag{44}$$

Using $-log(1-\theta) > \theta$ for $\theta \in (0,1)$ we can derive that (44) is guaranteed to hold whenever

$$\bar{k}\theta > \log \frac{n\mu_0}{\epsilon}.$$

By (43) it follows

$$\frac{1}{\theta} = \frac{4\sqrt{n} - 4\gamma + 2}{2 - (1+\beta)^2 - 9\gamma}$$

and it can be easily shown that $9\sqrt{n} > \frac{1}{\theta}$ whenever $2\beta + \beta^2 + 9\gamma < 1/3$. Then, $n\mu_k \leq \epsilon$ for any

$$k \geq 9\sqrt{n} \log \frac{n\mu_0}{\epsilon}.$$

Then, the stopping criterion is met after at most $18\sqrt{n} \log \frac{n\mu_0}{\epsilon}$ inexact Newton steps and (i) follows.

Moreover, by (26) we have:

$$S_{\bar{k},1} \bullet X_{\bar{k},2} \leq \mu_{\bar{k}} n \left(1 + \frac{\delta(S_{\bar{k},1}, \mu_{\bar{k}})}{\sqrt{n}}\right) \leq 3/2\epsilon,$$

as $\delta(S_{\bar{k},1}, \mu_{\bar{k}}) \leq 1/2$.

Then, Lemma 2 yields (ii).

□

We underline that, as already noted, at termination of Algorithm 4, a further outer iteration has to be carried out in order to recover primal feasibility. In this case, a sequence $\{S_{\bar{k}+1,\ell}\}_{\ell \geq 0}$ is generated and by Lemma 7 it follows that $\delta(S_{\bar{k}+1,\ell}, \mu_{\bar{k}+1})$ goes to zero whenever $\ell \to \infty$. Therefore, $\|r_{\bar{k}+1,\ell}\|$ goes to zero and $C \bullet X_{\bar{k}+1,\ell+1} - b^T y_{\bar{k}+1,\ell}$ is bounded by $3/2\epsilon$.

6 Matrix-free implementation

In this section we describe our matrix-free implementation of Algorithms 1 and 2.

Since we use a Krylov solver to compute an approximate solution of (9), the matrix M_ℓ given in (10) is required only to perform matrix-vector products. Due to the structure of M_ℓ, its action on a vector involves the inverse of the sparse matrix S_ℓ. Then, at each iteration of Algorithm 2, the Cholesky factorization $R_\ell^T R_\ell$ of the sparse matrix S_ℓ has to be computed. Taking into account that the dual matrix is assumed to be sparse, a sparse Cholesky factor is expected. Note that the structure of dual matrix does not change during the iterations of Algorithm 1, hence reordering of S_0 can be carried out once at the

very start of Algorithm 1 and then may be reused to compute the Cholesky factorization of S_ℓ at each iteration of Algorithm 2.

As pointed out in [2], the cost of evaluation of each column of M_ℓ depends strongly on the structure of constraint matrices A_i. More precisely, it is shown that the computation of the i-th column of M_ℓ involves p_i back-solves with S_ℓ, where p_i is the rank of the constraint matrix A_i. Then, if we assume that the constraint matrices have the same rank p, the cost of forming M_ℓ amounts to $2mp$ backsolves with R_ℓ. If M_ℓ is too large to be stored, we need to work in a matrix-free regime and therefore, in order to compute matrix vector products with M_ℓ we compute each column of M_ℓ once at a time and then discard it. In this latter case, letting $dens(R_\ell)$ be the density of the Cholesky factor R_ℓ, the cost of one matrix-vector product is given by $2mp \times O(n^2\, dens(R_\ell))$ operations. Clearly, this procedure allows to save memory as the whole matrix M_ℓ does not need to be stored, but it is more expensive than computing matrix vector-product with explicitly available matrix M_ℓ. Finally, we note that backsolves with R_ℓ can be performed in parallel.

More specifically, let us consider the constraint matrices represented as a sum of vector outer products, that is

$$A_i = \sum_{r=1}^{p_i} \alpha_{i,r} v_{i,r} v_{i,r}^T, \tag{45}$$

such that $\alpha_{i,r} \in \mathbb{R}$ and $v_{i,r} \in \mathbb{R}^n$, for $r = 1, \ldots, p_i$, e.g. represented by an eigendecomposition. Then a matrix-vector product can be performed using Algorithm 5 based on [2, Technique M2]. The algorithm is efficient when the ranks of the constraint matrices are low.

Algorithm 5 Matrix-vector product with the Schur complement matrix with constraint matrices in the form (45)

input: Cholesky factor $R_\ell \in \mathbb{R}^{n\times n}$ of S_ℓ and a vector $y \in \mathbb{R}^m$.
output: Matrix-vector product $q = M_\ell y$.
1: **for** $i = 1, \ldots, m$ **do**
2: **for** $r = 1, \ldots, p_i$ **do**
3: Solve the linear system $R_\ell^T u = v_{i,r}$.
4: Solve the linear system $R_\ell w = u$.
5: $q_i \leftarrow q_i + \alpha_{i,r}(w^T A_i w) * y_i$.
6: **end for**
7: **end for**
8: **return** q.

It is well known (see [23]) that a CG-like method applied to (9) may be slow, in particular in the late stage of an Interior Point method. On the other hand in our context traditional preconditioners cannot be incorporated as the matrix M_ℓ is dense and we assume that it is not available. For this latter reason matrix-free preconditioners are needed. Incomplete Cholesky (IC) factorizations are matrix-free in the sense that the columns of M_ℓ can be computed

one at a time, and then discarded, but they rely on drop tolerances to reduce fill-in and have unpredictable memory requirements. Alternative approaches with predictable memory requirements depend on the entries of M_ℓ, [17,19, 20,13] and have high storage requirements if M_ℓ is dense. On the other hand, the limited memory preconditioner given in [1,12] has a predictable memory requirements and for this reason we employed it in our runs. It consists in a partial Cholesky factorization limited to q columns of M_ℓ combined with a diagonal approximation of the Schur complement. In this approach, first an integer $q < m$ is chosen and the following formal partition of M_ℓ is considered

$$M_\ell = \begin{bmatrix} M_{11} & M_{21}^T \\ M_{21} & M_{22} \end{bmatrix},$$

where $M_{11} \in \mathcal{R}^{q\times q}$, $M_{21} \in \mathcal{R}^{(m-q)\times q}$, $M_{22} \in \mathcal{R}^{(m-q)\times(m-q)}$. Then, the first q columns of M_ℓ have to be computed and the Cholesky factorization of M_ℓ limited to

$$\begin{bmatrix} M_{11} \\ M_{21} \end{bmatrix}$$

is formed giving

$$M_\ell = \begin{bmatrix} L_{11} & \\ L_{21} & I_{m-q} \end{bmatrix} \begin{bmatrix} Q_{11} & \\ & Z \end{bmatrix} \begin{bmatrix} L_{11}^T & L_{21}^T \\ & I_{m-q} \end{bmatrix},$$

where $L_{11} \in \mathbb{R}^{q\times q}$ is lower triangular with ones on the diagonal, $L_{21} \in \mathbb{R}^{(m-q)\times q}$, $Q_{11} \in \mathbb{R}^{q\times q}$ is diagonal and

$$Z = M_{22} - M_{21}M_{11}^{-1}M_{21}^T,$$

is the Schur complement of M_{11} in M_ℓ. Then, letting $diag(\cdot)$ the operator that extracts the diagonal of a matrix and returns the diagonal matrix based on it, Z is approximated by

$$Q_{22} = diag(Z) = diag(M_{22}) - diag(L_{21}Q_{11}L_{21}^T)$$

and the following preconditioner is formed:

$$P_\ell = \underbrace{\begin{bmatrix} L_{11} & \\ L_{21} & I_{m-q} \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} Q_{11} & \\ & Q_{22} \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} L_{11}^T & L_{21}^T \\ & I_{m-q} \end{bmatrix}}_{L^T}.$$

Storage of the preconditioner requires remembering the partial Cholesky factor

$$L = \begin{bmatrix} L_{11} & \\ L_{21} & I_{m-q} \end{bmatrix}$$

and the diagonal Q, which overall needs at most $(q+1)m$ nonzero entries. The a priori known bound on the storage requirements is an advantage. To compute L one needs to compute the first q columns and the diagonal of M_ℓ

first. The cost of doing it is negligible compared to the cost of each matrix-vector product in the iterative linear solver. Therefore, the choice of q in this context depends only on the memory available.

Regarding spectral properties of the above outlined preconditioner, it has been proved in [1] that q eigenvalues of $P_\ell^{-1}M_\ell$ are equal to 1 and the remaining ones are the eigenvalues of $Q_{22}^{-1}Z$. Then,

$$\lambda_{max}(P_\ell^{-1}M_\ell) \leq trace(Q_{22}^{-1}Z) = m - q.$$

Therefore, the maximum eigenvalue of the preconditioned matrix stays bounded and does not grow to ∞ as μ_k goes to zero and the solution is approached. Moreover, in [12] a "greedy" heuristic technique acting on the largest eigenvalues of M_ℓ has been proposed. It consists in permuting rows and columns of M_ℓ so that M_{11} contains the q largest elements of the diagonal of M_ℓ. This choice is motivated by the well known result about the eigenvalues of the Schur complement (which was also used in [1])

$$\lambda_{max}(Q_{22}^{-1}Z) \leq \frac{\lambda_{max}(Z)}{\lambda_{min}(Q_{22})} \leq \frac{\lambda_{max}(M_{22})}{\lambda_{min}(Q_{22})} \leq \frac{trace(M_{22})}{\lambda_{min}(Q_{22})}.$$

Hence, by reducing the trace of M_{22}, a reduction in the value of $\lambda_{max}(P_\ell^{-1}M_\ell)$ is expected. Therefore we adopted this heuristic in our implementation.

As a final comment, the computation of $\delta(S_\ell, \mu)$ at Line 12 of Algorithm 2 is not needed. Indeed, as observed in [4],

$$\left\| S_\ell^{-1/2} \Delta S \, S_\ell^{-1/2} \right\|_F^2 = \Delta y^T M_\ell \Delta y.$$

Moreover, if Conjugate Gradient method is initialized with the zero vector and Δy is computed at a certain iteration of it then the quadratic form $\Delta y^T M_\ell \Delta y$ satisfies

$$\Delta y^T M_\ell \Delta y = \Delta y^T (b - \mu \mathcal{A}(S_\ell^{-1})).$$

A similar property holds also when the preconditioned CG is used [10].

7 Numerical experiments

In this section we report on our numerical experience with the proposed inexact dual-logarithmic barrier algorithms described in Algorithms 1-2 and 4. We did not use the backtracking strategy described in Algorithm 3 because we verified that in practice it was not needed to obtain convergence of Algorithm 2. We first describe the problem sets, then discuss the numerical results.

All the results have been obtained using a Matlab (R2015a) code on an Intel Core i5-6600K CPU 3.50 GHz x 4 16GB RAM.

7.1 Test problem sets

We evaluated the performance of the proposed methods on two classes of problems where the sparsity of the dual variable S is inherited from the sparsity of C and the structure of the A_i's. The first class of test examples arises in the SDP relaxation of maximum cut problems; the constraint matrices in this reformulation have rank 1 and $n = m$. The second class of test examples is obtained by a reformulation of matrix completion problems. The obtained SDP problems are characterized by constraint matrices of rank 2. In this case $n \ll m$ and therefore these problems could potentially be solved also by a primal-dual method. However, we considered this class for the sake of gaining more computational experience with problems which involve low-rank A_i's.

The maximum cut problem consists in partitioning the vertices of a graph into two sets that maximize the sum of the weighted edges connecting vertices in one set with vertices in the other. Its SDP relaxation [21] is

$$\begin{aligned} &\max\ C \bullet X \\ &\text{s.t.}\ \ diag(X) = e \\ &\qquad X \succeq 0, \end{aligned}$$

where e is the vector of all ones and C depends on the weighted adjacency matrix of the graph. Therefore, sparsity of C depends on the sparsity of the adjacency matrix of the graph. The dual problem is given by:

$$\begin{aligned} &\max\ e^T y \\ &\text{s.t.}\ \ Diag(y) + S = C \\ &\qquad S \succeq 0. \end{aligned}$$

The constraint matrices A_i are given by $A_i = e_i e_i^T,\ i = 1, \ldots, m$, where $e_i \in \mathbb{R}^m$ is the i-th vector of the canonical basis. Therefore, the A_is are trivially in the form (45), and each matrix has rank one. The form of matrix M_ℓ and vector $\mathcal{A}(S_\ell^{-1})$ in the right-hand side of (9) simplifies and is given by

$$M_\ell = (S_\ell^{-1}) .* (S_\ell^{-1}) \ \text{ and } \ \mathcal{A}(S_\ell^{-1}) = diag(M_\ell),$$

where we have borrowed the Matlab notation; i.e. M_ℓ is the componentwise product of (S_ℓ^{-1}) times itself. We specialized Algorithm 5 for the maximum cut problem in Algorithm 6, taking into account the special structure of each A_i.

As a second set of problems, we considered large and sparse SDPs which originate from a reformulation of the matrix completion problem, that is the problem of recovering a low rank data matrix $B \in \mathbb{R}^{\hat{n} \times \hat{n}}$ from a sampling of its entries [5].

Let $B_{s,t}, (s,t) \in \Omega$ be the given entries of matrix B. The SDP relaxation of this problem is given by

$$\begin{aligned} &\min\ Tr(W_1) + Tr(W_2) \\ &\text{s.t.}\ \ \begin{bmatrix} W_1 & \bar{X} \\ \bar{X}^T & W_2 \end{bmatrix} \succeq 0 \\ &\qquad \bar{X}_{s,t} = B_{s,t} \quad (s,t) \in \Omega, \end{aligned} \tag{46}$$

Algorithm 6 Matrix-vector product with the Schur complement matrix in the maximum cut problem

input: Cholesky factor $R_\ell \in \mathbb{R}^{n\times n}$ of S_ℓ and a vector $y \in \mathbb{R}^n$.
output: Matrix-vector product $q = M_\ell y$.
1: **for** $j = 1, \dots, n$ **do**
2: Solve the linear system $R_\ell^T u = e_j$.
3: Solve the linear system $R_\ell w = u$.
4: $q_j \leftarrow \sum_{i=1}^n w_i^2 y_i$.
5: **end for**
6: **return** q.

where $\bar{X}, W_1, W_2 \in \mathbb{R}^{\hat{n}\times\hat{n}}$ are the unknowns, and $B_{s,t}, (s,t) \in \Omega$ are given, see [18].

We can reformulate (46) as (1) setting $n = 2\hat{n}$ and m equal to the cardinality of Ω, i.e. the number of known entries of B. The primal variable X takes the form $X = \begin{bmatrix} W_1 & \bar{X} \\ \bar{X}^T & W_2 \end{bmatrix} \in \mathbb{R}^{n\times n}$, while the matrix C in the objective is $C = \frac{1}{2} I_n$. In order to define the operators A_i's, we enumerate the m couples $(s,t) \in \Omega$ so that $b_i = B_{s,t}$ for $i \in \{1, \dots m\}$ and $(s,t) \in \Omega$. Moreover, for each $(s,t) \in \Omega$ let us define the matrix $\Theta^{st} \in \mathbb{R}^{\hat{n}\times\hat{n}}$ such that

$$(\Theta^{st})_{kl} = \begin{cases} 1 & \text{if } (k,l) = (s,t) \\ 0 & \text{otherwise.} \end{cases}$$

Then, the constraint matrices $A_i \in \mathbb{R}^{n\times n}$ with $i \in \{1, \dots m\}$ (and corresponding $(s,t) \in \Omega$) are given by

$$A_i = \frac{1}{2} \begin{bmatrix} 0 & \Theta^{st} \\ (\Theta^{st})^T & 0 \end{bmatrix}. \tag{47}$$

Having defined all the ingredients C, b, A_i, X as above, we obtain the SDP relaxation of matrix completion problem in the form (1).

We note that since C is diagonal and A_i's have at most 2 nonzero elements, the slack variable S has at most $2m + n$ nonzero elements. We also underline that $m \ll \hat{n}^2$.

Constraint matrices are rank 2 and they can easily be expressed in the form (45) using the eigendecomposition of the 2×2 matrix $\begin{bmatrix} 0 & 0.5 \\ 0.5 & 0 \end{bmatrix}$. This is described in Algorithm 7, where we borrowed again the Matlab notation.

7.2 Implementation issues for the long-step algorithm

We set the initial parameters $\mu = 1$ in Algorithm 1. In all the iterations except for the last one, the Newton procedure was stopped whenever a full Newton step is taken (i.e. $\alpha = 1$ in Line 11 of Algorithm 2) and $\delta(S_\ell, \mu) \le 0.5$. Then, only a rough accuracy is required in the solution of the nonlinear systems (3)

Algorithm 7 Matrix-vector product with the Schur complement matrix in the matrix completion problem

input: Cholesky factor $R_\ell \in \mathbb{R}^{n\times n}$ of S_ℓ, eigenvalues $\bar\lambda_1, \bar\lambda_2$ and eigenvectors $\bar v_1, \bar v_2$ of $\begin{bmatrix} 0 & 0.5 \\ 0.5 & 0 \end{bmatrix}$ and a vector $y \in \mathbb{R}^m$.
output: Matrix-vector product $q = M_\ell y$.
1: **for** $i = 1, \ldots, m$ **do**
2: **for** $r = 1, 2$ **do**
3: Let (k, l) be the row and column indexes of the nonzero element of $A_i(1 : n/2, :)$.
4: Set $z \leftarrow$ `zeros`$(n, 1)$, $z(k) \leftarrow \bar v_r(1)$, $z(l) \leftarrow \bar v_r(2)$;
5: Solve the linear system $R_\ell^T u = z$.
6: Solve the linear system $R_\ell w = u$.
7: $q_i \leftarrow q_i + \lambda_r (w^T A_i w) * y_i$.
8: **end for**
9: **end for**
10: **return** q.

except in the last iteration where, in order to recover primal feasibility, the Newton process is carried out until $\delta(S_\ell, \mu) \le 10^{-5}$.

In Algorithm 2 we employed the Matlab function `pcg` to compute the approximate solution of (9), i.e. we used the CG method [11]. We work in a nearly matrix-free regime, so we do not store the whole matrix M_ℓ but we only store the q columns needed for building the preconditioner. At Line 4 of Algorithm 2, we set $\eta_\ell = 10^{-3}$ for each ℓ, that is CG is stopped when the relative residual is less than 10^{-3}. However, if (16) is not satisfied within 100 iterations, CG is halted and the algorithm progresses with the best iterate computed by CG, that is the approximation returned by `pcg` at which the smallest residual has been obtained. Denoting with `CGit` the CG iteration at which this occurred, we set `CGit` as a limit on the number of CG iterations at the subsequent Newton's iteration.

Finally, at Line 8 of Algorithm 2 we perform the Cholesky decomposition of $S_\ell + \alpha \Delta S$ in order to detect whether the matrix is positive definite.

7.3 Numerical results for the long-step algorithm: maximum cut test problems

We considered random graphs available in the GSet group of University of Florida Sparse Matrix Collection [7]. More precisely, we selected five matrices (`G48, G57, G62, G65, G66, G67`) corresponding to toroidal graphs of dimension that ranges from $m = 3000$ to $m = 20000$ and two matrices (`G60, G63`) of dimension $m = 7000$ corresponding to less structured graphs. We also considered the smaller problem `G11` to show in detail the preconditioner behaviour.

In Table 1 we report statistics of our runs. These results have been obtained setting $\epsilon = 10^{-8}$ in Algorithm 1 and computing $q = 0.3m$ columns in the partial Cholesky preconditioner, thus saving 70% of memory in comparison with a direct approach which applies the complete Cholesky factorization of

Test name	m	$dens(S)$	$dens(R)$	IT_NEW	CG_AV	TIME_AV	$\|\mathcal{A}(X)-b\|$	$X \bullet S$
G48	3000	1.7e-3	1.7e-2	48	7	1.2e1	1.3e-8	3.0e-5
G57	5000	1.0e-3	9.0e-3	67	32	1.2e2	1.9e-6	1.9e-5
G60	7000	8.4e-4	4.2e-2	108	12	2.1e2	1.9e-6	4.1e-4
G62	7000	7.1e-4	7.1e-3	73	37	3.1e2	3.5e-6	2.1e-4
G63	7000	1.8e-3	5.6e-2	188	8	2,7e2	2.5e-6	5.2e-4
G65	8000	6.2e-4	7.1e-3	72	36	3.5e2	1.8e-6	1.7e-4
G66	9000	5.5e-4	3.4e-3	71	41	6.5e2	1.5e-6	1.3e-4
G67	10000	5.0e-4	6.3e-3	71	42	9.3e2	1.2e-6	1.0e-4

Table 1 Statistics of the runs on the GSET family

the Schur complement. Problems corresponding to toroidal graphs have been solved using $\sigma = 0.1$, while results for the graphs G60 and G63 were obtained using $\sigma = 0.5$.

A feasible starting couple (S, y) is easily obtained as in [24].

The heading of the columns has the following meaning: $dens(S)$: density of the dual matrix S_0; $dens(R)$ density of the Cholesky factor of S_0; IT_NEW: overall number of inner Newton iterations; CG_AV: average number of CG iterations for each Newton iteration; TIME_AV: average time in seconds to perform one inner Newton iteration; $\|\mathcal{A}(X) - b\|$: primal feasibility; $X \bullet S$: complementarity gap. We can observe that both S_0 and its Cholesky factor are quite sparse in all the tests. The number of overall Newton iterations and the average number of CG iterations are more or less the same in problems G57, G62, G65, G66 and G67. So they do not seem to increase with m. The higher number of Newton iterations in G60 and G63 is due to the less aggressive choice of σ. Problem G48 is easier than the other ones as it requires noticeably fewer Newton and CG iterations. As expected the execution time increases with the dimension of the problem. It should be underlined that the high execution time is the price we pay for avoiding to store matrix M_ℓ. For example, in the solution of problem G67 the average time to perform one inner Newton iteration drops to 111 seconds if matrix M_ℓ is computed once and then stored. Moreover, we observe that a major part of the execution time originates from the last iteration of Algorithm 1 as the linear systems become more difficult and their solution requires a higher number of CG iterations and therefore a higher number of matrix-vector products with M_ℓ.

To give some insight into the behaviour of the preconditioner, we considered the smaller problem G11 (m=800) and we focused on the most challenging last interior point iteration where the arising linear systems are ill-conditioned and several inexact Newton steps are required because the primal feasible solution X is sought. For this example, 8 Newton iterations were needed; we built the sequence of linear systems generated by Algorithm 2 (with the partial Cholesky preconditioner using the 30% of the columns of M) and we solved it also with unpreconditioned CG and with CG using the Diagonal preconditioner.

Firstly, we compare in Table 2 the smallest and largest eigenvalues of matrix M_ℓ with those of M_ℓ preconditioned by partial Cholesky ($P^{-1}M$) and

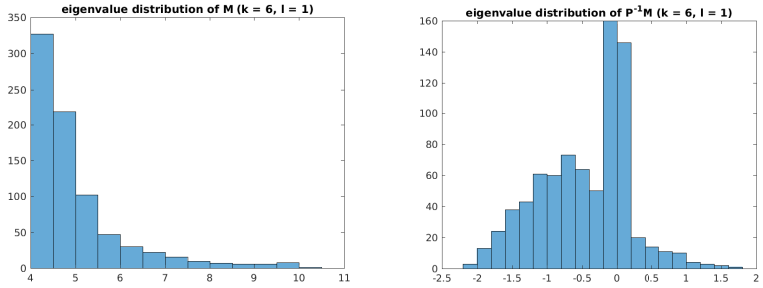

Fig. 1 Eigenvalue distribution of M and $P^{-1}M$ at the first Newton's iteration (last outer iteration).

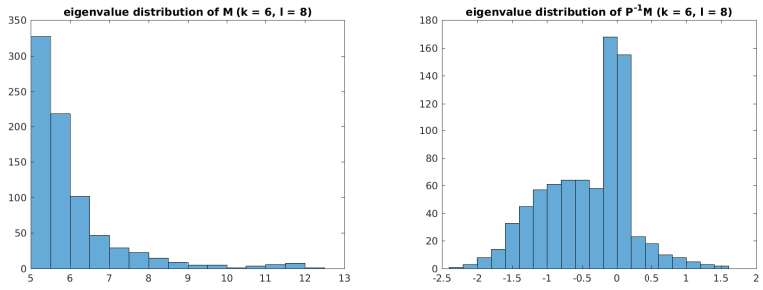

Fig. 2 Eigenvalue distribution of M_ℓ and $P^{-1}M_\ell$ at the last Newton's iteration (last outer iteration).

M_ℓ preconditioned by the Diagonal preconditioner ($D^{-1}M$). We note that the eigenvalues of unpreconditioned M_ℓ vary between 10^4 and 10^{10} at iteration 1 and their spread increases to $10^4 - 10^{12}$ at iteration 8. The spread of eigenvalues of the preconditioned Schur complement $P_\ell^{-1}M_\ell$ is significantly smaller. Indeed, the eigenvalues vary between 10^{-3} and 10^1, that is the partial Cholesky preconditioner drastically reduces the largest eigenvalues of the unpreconditioned matrix M_ℓ; the smallest eigenvalue is pushed closer to zero but overall the condition number of the preconditioned matrix is smaller than that of M_ℓ. On the other hand, the Diagonal preconditioner is not as effective. It simply shifted the whole spectrum towards zero. We also plot in Figures 1 and 2 the histograms of eigenvalues of M_ℓ and $P_\ell^{-1}M_\ell$ of the first ($\ell = 1$) and the last ($\ell = 8$) Newton system. The histograms use the logarithmic scale to demonstrate the magnitude of eigenvalues. They reveal that in both linear systems more than 50% of the eigenvalues of $P_\ell^{-1}M_\ell$ are clustered around one. Although a good clustering of the spectrum of $P_\ell^{-1}M_\ell$ does not necessarily imply fast convergence of the conjugate gradient method (see Section 5.6.5 in [16]) it usually benefits its behaviour.

Secondly, from the statistics reported in Table 3, we can observe that CG preconditioned by the partial Cholesky (`CG+P`) outperforms the unpreconditioned CG. Failures are denoted by the symbol '*' and, in this case, the min-

ℓ	$\lambda_{min}(M)$	$\lambda_{max}(M)$	$\lambda_{min}(D^{-1}M)$	$\lambda_{max}(D^{-1}M)$	$\lambda_{min}(P^{-1}M)$	$\lambda_{max}(P^{-1}M)$
1	1.2e4	1.7e10	1.2e-4	1.7e2	8.2e-3	4.7e1
2	2.9e4	1.0e11	4.7e-5	1.7e2	1.0e-2	2.8e1
3	1.7e4	3.7e12	8.2e-6	1.8e2	4.1e-3	3.8e1
4	1.3e4	2.2e12	1.1e-5	1.8e2	4.0e-3	3.9e1
5	1.2e4	1.8e12	1.2e-5	1.8e2	3.5e-3	3.8e1
6	1.2e4	1.8e12	1.2e-5	1.8e2	6.8e-4	6.0e1
7	1.2e4	1.8e12	1.2e-5	1.8e2	1.1e-3	3.6e1
8	1.2e4	1.8e12	1.2e-5	1.8e2	1.4e-3	5.7e1

Table 2 Problem `G11`: minimum and maximum eigenvalues of M, M preconditioned by the diagonal (D) and the partial Cholesky (P) preconditioner, at the last outer iteration of Algorithm 1

ℓ	`CG`	`CG + D`	`CG + P`	$\|r_\ell^{\texttt{CG}}\|$	$\|r_\ell^{\texttt{CG+D}}\|$	$\|r_\ell^{\texttt{CG+P}}\|$
1	*	*	6	1.1e-3	1.1e-3	
2	*	*	12	1.1e-3	1.0e-3	
3	54	30	17			
4	*	*	23	1.4e-3	1.4e-3	
5	*	*	66	4.0e-3	4.0e-3	
6	*	*	31	1.1e-2	1.1e-2	
7	*	*	*	2.3e-2	2.3e-2	2.6e-3
8	*	*	*	3.5e-1	3.5e-1	3.1e-3

Table 3 Problem `G11`: number of CG iterations (unpreconditioned and preconditioned with D and P) and norm of the corresponding residual, at the last outer iteration of Algorithm 1.

imum value of the relative residual obtained during the iterations performed by CG is also reported. `CG+P` fails in the last two inner iterations but the obtained residuals are quite close to the desired values, while unpreconditioned CG stops with large residuals. Unsurprisingly, CG with Diagonal preconditioner (`CG+D`) behaves as poorly as the unpreconditioned CG. We also stress that the cost of application of the partial Cholesky preconditioner is negligible in the tested matrix-free implementation compared to the cost of performing matrix-vector product with M_ℓ and therefore the preconditioner choice has to be guided only by its behavior in terms of reducing CG iterations.

Finally, in Figure 3 we plot the convergence history of the whole procedure applied to problem `G48`. In particular the value of the duality gap and of primal feasibility versus the Newton iterations are displayed. We can observe that the duality gap reduces with μ and primal feasibility is recovered at the last outer iteration.

7.4 Numerical results for the long-step algorithm: matrix completion test problems

In the solution of the SDP reformulation of matrix completion problems, the extra-diagonal submatrix $\bar{X}$ of the primal variable X returned by Algorithm 1 is the computed approximation of matrix B that we would like to recover.

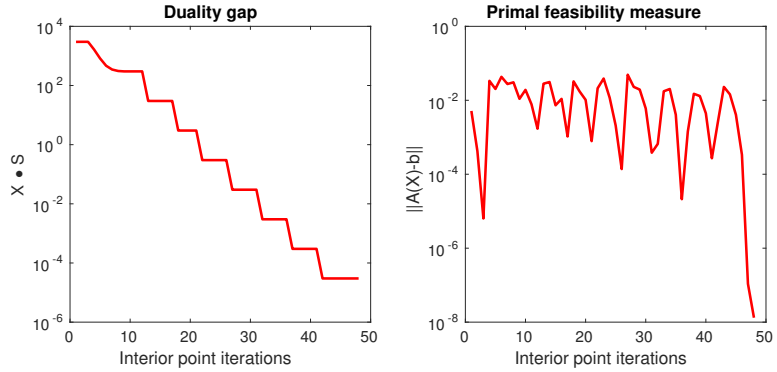

Fig. 3 Problem G48: duality gap and primal feasibility.

Test	r	n	m	IT_NEW	CG_AV	TIME_AV	ϵ_B	$\|\mathcal{A}(X)-b\|$	$X \bullet S$
MC1	4	800	12736	76	52	2.3e2	5e-4	4.3e-3	6.1e-3
MC2	5	1000	19900	81	37	5.1e2	4e-6	6.4e-6	7.6e-3
MC3	6	1200	28656	82	36	9.8e2	2e-6	4.8e-6	9.2e-3
MC4	7	1400	39004	80	33	1.0e3	1e-6	2.1e-6	1.0e-2
MC5	8	1600	50944	79	33	3.2e3	8e-7	1.0e-6	1.2e-2

Table 4 Statistics of the runs on the matrix completion set.

Following [5] we considered a matrix recovered when $\epsilon_B = \|\bar{X} - B\|_F / \|B\|_F \leq 10^{-3}$ and we observed that high accuracy in the solution of the SDP problem is not needed to get such accuracy in the recovered matrix. Then, we set the stopping tolerance ϵ equal to 10^{-6} and $\sigma = 0.5$ in Algorithm 1.

Following [5], we generated matrices $B \in \mathbb{R}^{\hat{n}\times\hat{n}}$ of rank r, by sampling two $\hat{n}\times r$ factors B_L and B_R, each having independently and identically distributed Gaussian entries, and setting $B = B_L B_R^T$. The set of observed entries Ω is generated sampling m entries of B uniformly at random. We set $m = 4r(2\hat{n} - r)$, that is m is four times the degrees of freedom of rank r matrices and we choose $r = \hat{n}/100$. This way, we obtained dual matrices S with density $dens(S) \approx 4 \cdot 10^{-2}$ and Cholesky factor's density of the order of $1.4 \cdot 10^{-1}$. The dimensions of the generated SDP problems are shown in Table 4.

Since in these problems m is large and reaches tens of thousand, we set $q = 0.1m$ in the partial Cholesky preconditioner with a saving of the 90% of memory. A feasible starting couple (S, y) is trivially computed by setting $y = 0$ and $S = C$.

Results are reported setting the Matlab random generator `rng('default')` and `rng(1)`, similar results have been found with different random seeds. In Table 4 we report statistics of our runs. The headings are the same as in Table 1 with extra information on the rank r of matrix B to be recovered and the relative error of the recovered matrix ϵ_B.

We observe that the average number of nonlinear Newton iterations is larger than in maximum cut problems indicating that the arising nonlinear systems are quite difficult. On the other hand the average number of CG iterations remains small. Overall, the average CPU time is comparable to that obtained for maximum cut problems of similar size. The matrices B are recovered with a satisfactory accuracy.

7.5 Numerical validation of the short-step algorithm

Following the suggestion of the Anonymous Referee we carried experiments in order to validate the theoretical analysis conducted in Section 5 for the Short-Step Inexact Dual-Logarithmic barrier algorithm. We implemented Algorithm 4 and chose parameters β, γ, θ according to (43) and satisfying the assumptions in Theorem 1. We set the tolerance $\epsilon = 10^{-5}$. After iteration $\bar{k}$, at which the algorithm stops, a further outer iteration is performed until $\delta(S_{\bar{k}+1,\ell}, \mu_{\bar{k}+1}) \leq 10^{-5}$. Finally, we set a maximum number of 50 CG iterations. Algorithm 4 requires a starting couple (S, μ) such that S is feasible, $\mu > 0$ and $\delta_{ex}(S, \mu) \leq 1/2$. Setting $\mu = 1$, we built S by applying one iteration of Algorithm 1. Linear systems at Lines 5-6 of Algorithm 2 have been solved until $\delta_{ex}(S_\ell, \mu) \leq 1/2$ ($\tau = 1/2$).

Here we report the results obtained in the solution of the maximum cut problem using graph G11. Using the above strategy, we obtained for G11 a well-centered dual feasible S with $\delta_{ex}(S, 1) = 0.1$. For this problem $n = 800$ and consequently we used $\theta = 0.008$ with $\beta = \gamma = 0.01$. At the last interior point iteration, three Newton steps have been performed. Note that for this problem $\|A^\dagger\| = 1$.

In Figures 4 and 5 we report statistics concerning the first and last 5 outer iterations, each involving 2 Newton inner iterations, except for the last one where 3 Newton iterations have been performed. In Figure 4, we plot the values of $\log_{10} \delta(S_{k,\ell}, \mu_k)$; the horizontal line corresponds to the value $\log_{10}(1/2)$. We can observe that, as Lemma 10 states, starting from $(S_{k,0}, \mu_k)$ such that $\delta_{ex}(S_{k,0}, \mu_k) \leq 1/2$ after merely 2 Newton steps we obtain again a starting value $S_{k+1,0}$ for the next iteration that is well-centered with respect to μ_{k+1}.

Let $\rho_{k,\ell}$ for $\ell = 0, 1$ be the upper bound on the relative residual norms in Algorithm 4 (see (32) and (34)). Figure 5 shows the relative accuracy requirement $\rho_{k,\ell}$ dictated by the theory and the actual value of the relative residual $\|r_{k,\ell}\|/\|b/\mu_k - \mathcal{A}(S_{k,\ell}^{-1})\|$ returned by CG, still in the first and in the last 5 outer iterations. We observe that in the first iterations, the two quantities have the same order of magnitude that is around 10^{-1} and 10^{-7} in the first and the second Newton iteration, respectively. On the other hand, in the last iterations the prescribed accuracy is too tight and it is not needed to preserve the two-step quadratic convergence of the method. In fact, we can observe from the figures that in 50 iterations CG provides a higher value of the relative residual than that prescribed by the theory, but after 2 Newton steps, we still obtain a

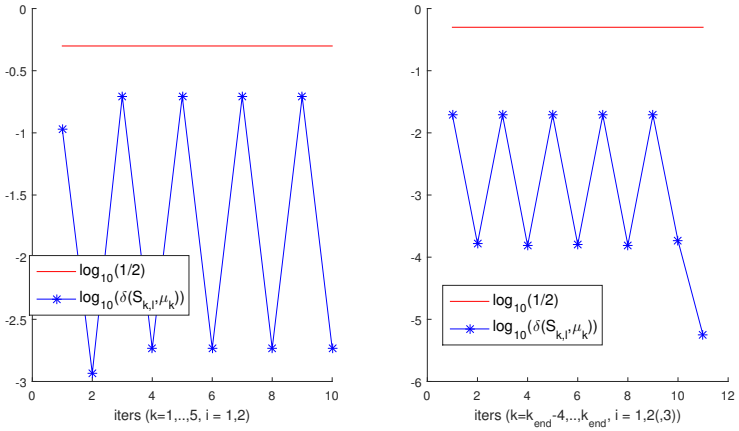

Fig. 4 Problem G11: Values of $\delta(S_{k,\ell}, \mu_k)$ for $\ell = 1, 2(, 3)$ in first 5 (left figure) and last 5 (right figure) outer iterations (logarithmic scale).

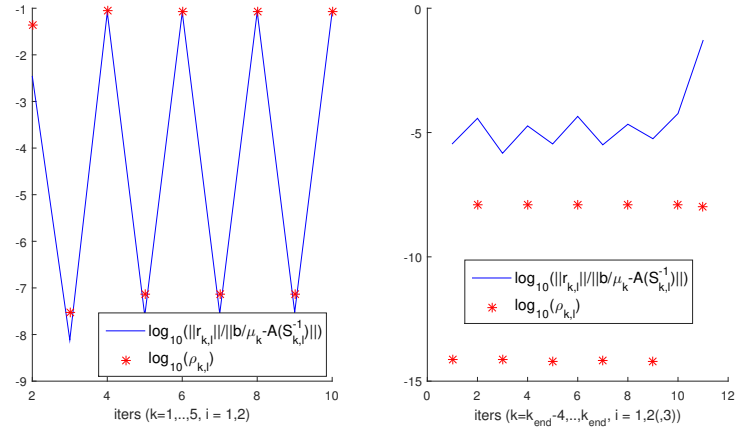

Fig. 5 Problem G11: Values of $\rho_{k,\ell}$ and $\|r_{k,\ell}\|/\|b/\mu_k - \mathcal{A}(S_{k,\ell}^{-1})\|$ for $\ell = 1, 2(, 3)$ in first 5 (left) and last 5 (right) outer iterations (logarithmic scale).

starting value $S_{k+1,0}$ for the next iteration that is well-centered with respect to μ_{k+1}, as already noted.

8 Conclusions

A variant of dual interior point method for semidefinite programming has been proposed in this paper. The method uses inexact Newton steps and it is well suited to problems with sparse dual matrix S. The computations avoid any ex-

plicit use of primal variable X (which might be noticeably more dense than S) and therefore the method offers advantages in memory required to solve very large SDPs. Krylov subspace method preconditioned with partial Cholesky factorization of the Schur complement matrix is employed to solve the reduced KKT systems. Convergence properties and the $\mathcal{O}(\sqrt{n}\ln\frac{n}{\varepsilon})$ complexity result have been established. A prototype Matlab-based implementation of the method has been developed and it has been demonstrated to perform well on medium scale SDP problems arising in maximum cut and matrix completion problems.

Acknowledgements The authors would like to thank the anonymous referees for their suggestions, which lead to significant improvement of the manuscript.

References

1. Bellavia, S., Gondzio, J., Morini, B.: A matrix-free preconditioner for sparse symmetric positive definite systems and least-squares problems. SIAM Journal on Scientific Computing **35**, A192–A211 (2013)
2. Benson, S.J., Ye, Y.: Algorithm 875: DSDP5-software for semidefinite programming. ACM Transaction on Mathematical Software **34**(3), Art.16,20 (2008)
3. Benson, S.J., Ye, Y., Zhang, X.: Mixed linear and semidefinite programming for combinatorial and quadratic optimization. Optimization Methods and Software **11**(1-4), 515–544 (1999)
4. Benson, S.J., Ye, Y., Zhang, X.: Solving large-scale sparse semidefinite programs for combinatorial optimization. SIAM Journal on Optimization **10**(2), 443–461 (2000)
5. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. Foundations of Computational Mathematics **9**(6), 717–772 (2009)
6. Choi, C., Ye, Y.: Solving large-scale sparse semidefinite programs using the dual scaling algorithm with an iterative solver. Tech. rep. (2000)
7. Davis, T.A., Hu, Y.: The University of Florida sparse matrix collection. ACM Transactions on Mathematical Software **38**(1), 1–25 (2011)
8. De Klerk, E.: Aspects of semidefinite programming: interior point algorithms and selected applications, vol. 65. Springer Science & Business Media (2006)
9. Eisenstat, S.C., Walker, H.F.: Globally convergent inexact Newton methods. SIAM Journal on Optimization **4**(2), 393–422 (1994)
10. Fountoulakis, K., Gondzio, J.: A second-order method for strongly convex l1-regularization problems. Mathematical Programming A **156**(1), 189–219 (2016)
11. Golub, G.H., Van Loan, C.: Matrix Computations, (2nd ed.). The Johns Hopkins University Press (1989)
12. Gondzio, J.: Matrix-free interior point method. Computational Optimization and Applications **51**, 457–480 (2012)
13. Gould, N., Scott, J.A.: The state-of-the-art of preconditioners for sparse linear least-squares problems. Tech. Rep. RAL-P-2015-010, Rutherford Appleton Laboratory, Chilton, England (2015)
14. Helmberg, C., Rendl, F.: A spectral bundle method for semidefinite programming. SIAM Journal on Optimization **10**, 673–696 (2000)
15. Kocvara, M., Stingl, M.: On the solution of large-scale SDP problems by the modified barrier method using iterative solvers. Mathematical Programming **109**, 413–444 (2007)
16. Liesen, J., Strakos, Z.: Krylov Subspace Methods: Principles and Analysis. Oxford University Press (2012)
17. Lin, C.J., Morè, J.J.: Incomplete Cholesky factorizations with limited memory. SIAM Journal on Scientific computing **21**, 24–25 (1999)
18. Recht, B., Fazel, M., Parrilo, P.A.: Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. SIAM Review **52**(3), 471–501 (2010)

19. Scott, J.A., Tuma, M.: HSL_MI28: An efficient and robust limited-memory incomplete Cholesky factorization code. ACM Transactions on Mathematical Software **40**, Art.24,19 (2014)
20. Scott, J.A., Tuma, M.: On positive semidefinite modification schemes for incomplete Cholesky factorization. SIAM Journal on Scientific computing **36**, A609–A633 (2014)
21. Todd, M.J.: Semidefinite optimization. Acta Numerica 2001 **10**, 515–560 (2001)
22. Toh, K.C.: Solving large scale semidefinite programs via an iterative solver on the augmented systems. SIAM Journal on Optimization **14**, 670–698 (2004)
23. Toh, K.C., Kojima, M.: Solving some large scale semidefinite programs via the conjugate residual method. SIAM Journal on Optimization **12**(3), 669–691 (2002)
24. Tütüncü, R.H., Toh, K.C., Todd, M.J.: Solving semidefinite-quadratic-linear programs using SDPT3. Mathematical programming **95**(2), 189–217 (2003)
25. Vandenberghe, L., Andersen, M.S.: Chordal graphs and semidefinite optimization. Foundation and Trends in Optimization **1**(4), 241–433 (2015)
26. Vandenberghe, L., Boyd, S.: Semidefinite programming. SIAM Review **38**(1), 49–95 (1996)
27. Zhao, X.Y., Sun, D., Toh, K.C.: A Newton-CG augmented Lagrangian method for semidefinite programming. SIAM Journal on Optimization **20**(4), 1737–1765 (2010)